LIST OF FIGURES

LIST OF TABLES

1.0 INTRODUCTION

LDS-Akron has developed a 3D Laser Radar Vision Processor system capable of detecting, classifying, and identifying small mobile targets as well as larger fixed targets using 3-dimensional laser radar imagery for use with a robotic type system. This processor system is designed to interface with NASA Johnson Space Center in-house EVA Retriever robot program and provide to it needed information so it can fetch and grasp targets in a space-type scenario.

2.0 HARDWARE DESCRIPTION

The 3D Laser Radar Vision Processor system is an IBM-XT compatible computer with an INMOS board containing four transputers inserted in one of the IBM-XT expansion slots. This hardware is illustrated in Figure 1. The logical connection of the hardware is shown in Figure 2.

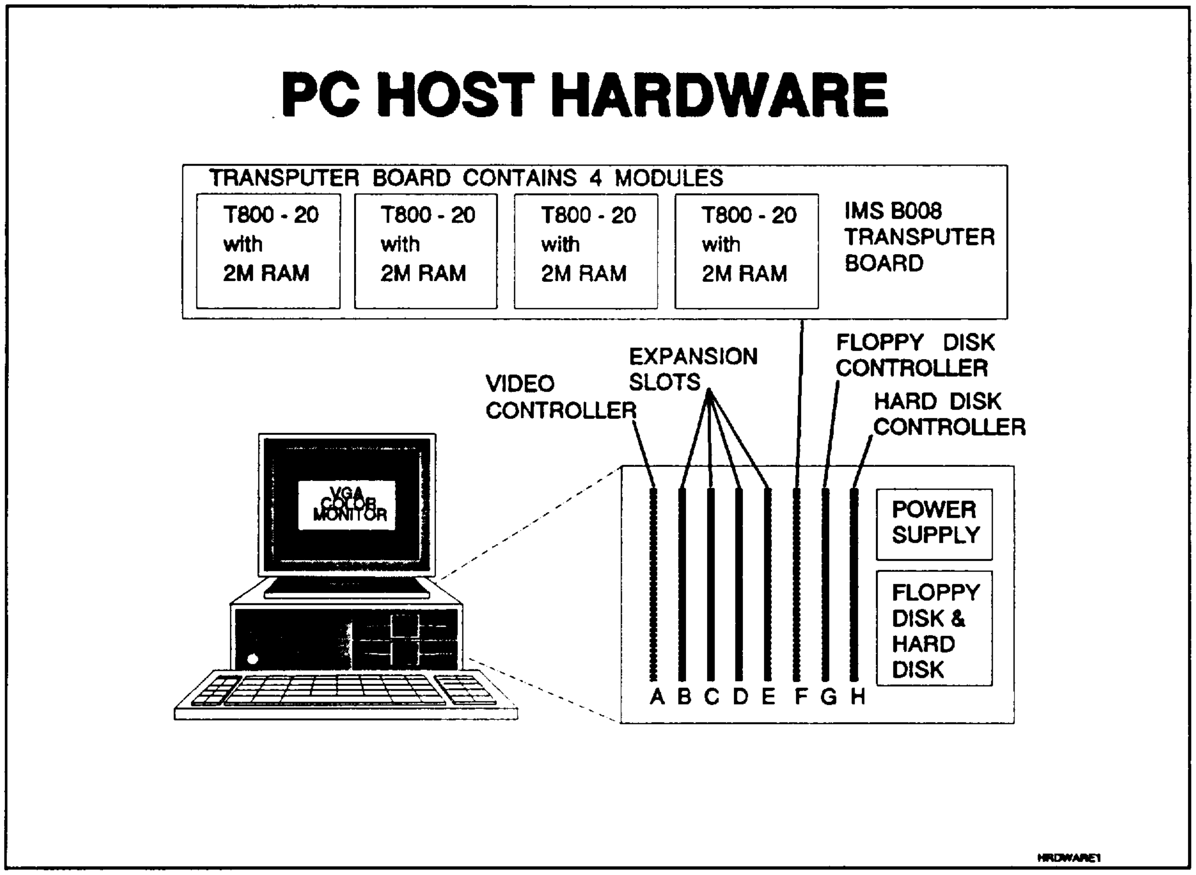


Figure 1. PC Host Computer

The delivered hardware consists of the following items:

1. IBM-XT compatible computer
2. Transputer add-in board for the IBM-XT
3. VGA monitor and card

The computer system is an IBM-XT compatible computer which serves as a stand-alone development system for the 3D Laser Radar Vision Processor. This computer includes the following: a 10Mhz XT motherboard, 40 MBytes hard disk, one 360K floppy, case, 150W power supply, keyboard, VGA monitor, and VGA display card.

The heart of the Loral's 3D Laser Radar Vision Processor is an IMSB008 Transputer board populated with four IMSB404 modules. The IMSB008 Transputer board is an add-in board for the IBM PC, which takes up one slot in the PC and provides support for up to ten INMOS Transputer modules. This support includes a communication link between the XT and the transputer's network and the interconnection network between the transputers. The transputer interconnection network is provided by an on-board IMS C004 link switch. The IMS C004 allows the user to specify transputer interconnections without doing any physical wiring. Controlling the IMS C004 is an on-board T212 processor.

The transputer module that will be used with the IMSB008 Transputer board is the IMS404 module. The IMS404 module contains one 20 MHz INMOS T800 Transputer along with 2 MBytes of dynamic RAM memory. The T800 transputer is a 32-bit floating point RISC processor. An integral part of the T800 Transputer is its ability to communicate with up to four other transputers via high speed (2.35 MBytes) serial link.

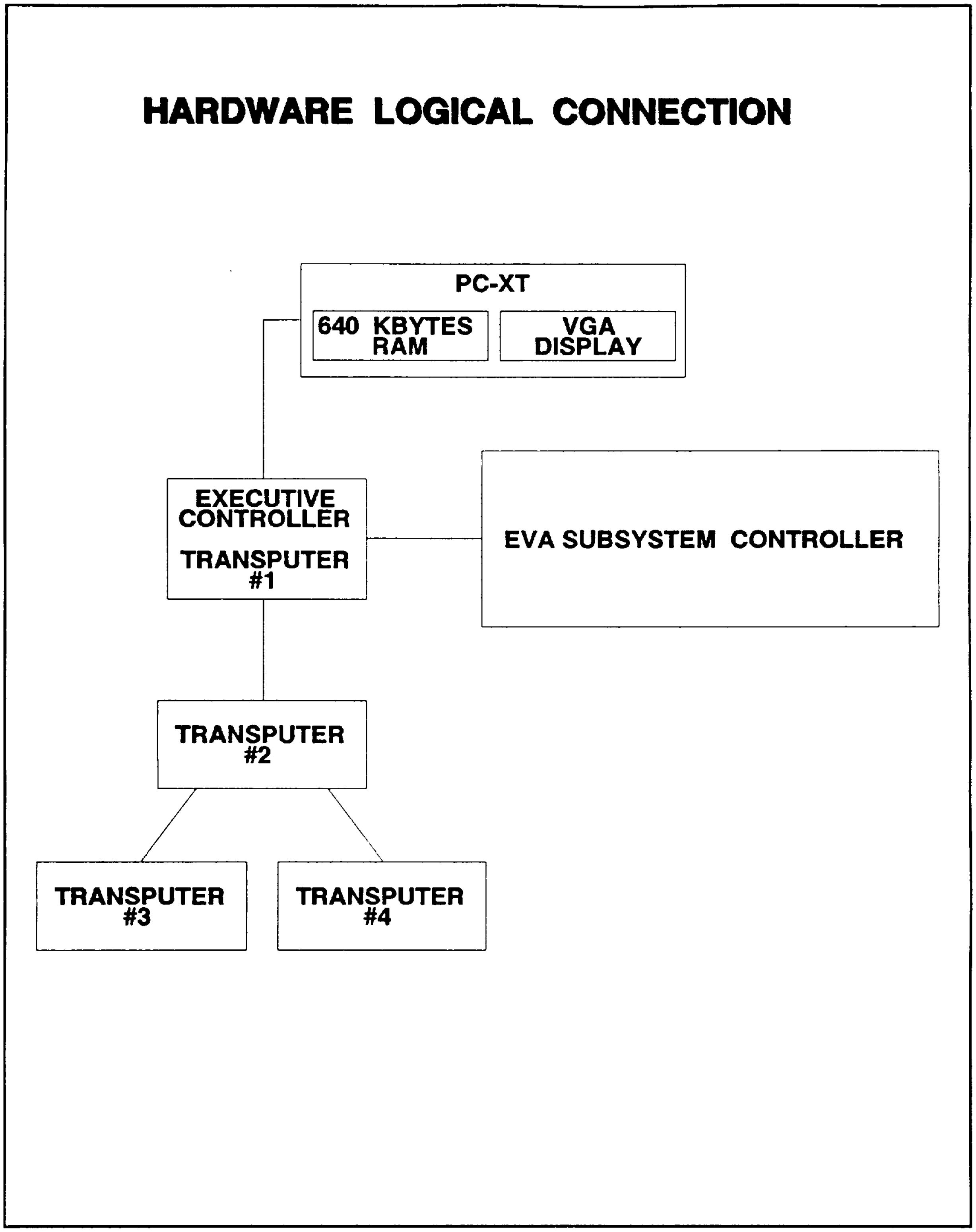


Figure 2. Hardware Logical Connection

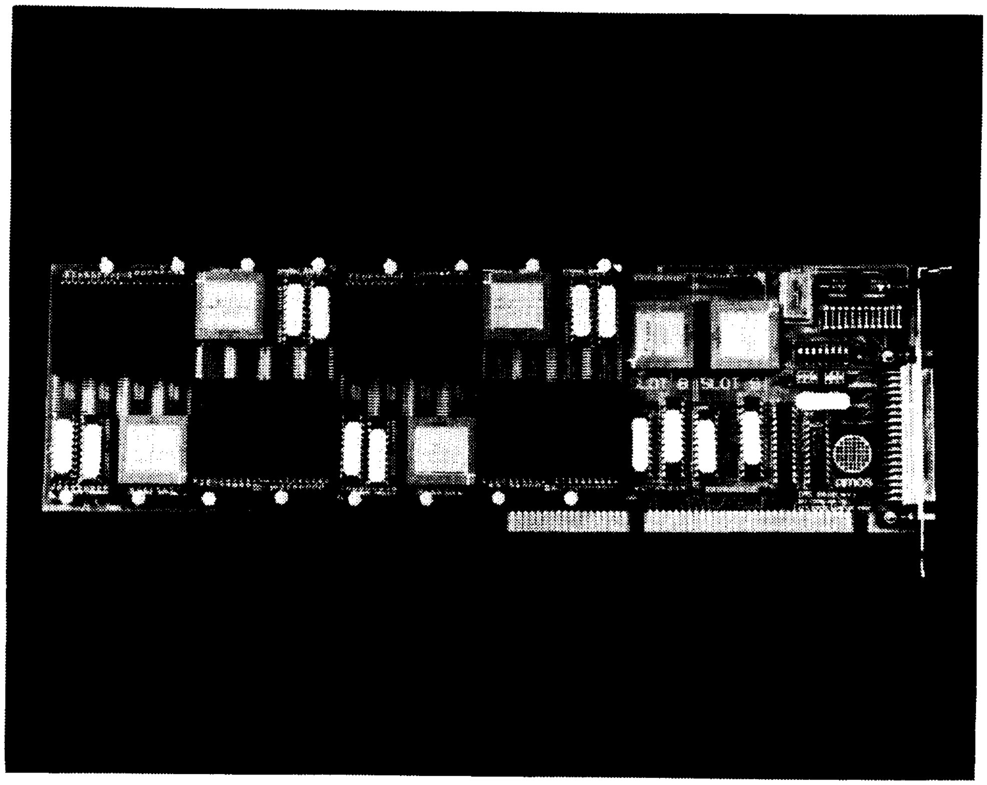


Figure 3. B008 Transputer Board

3.0 PROGRAM OPERATION

3.1 Program Execution

In order to run the NASA 3-D Laser vision processor, the default directory must be C:\NASA, which contains the program source, batch, and executable files. To start the program, type:

C:\NASA> **gn**

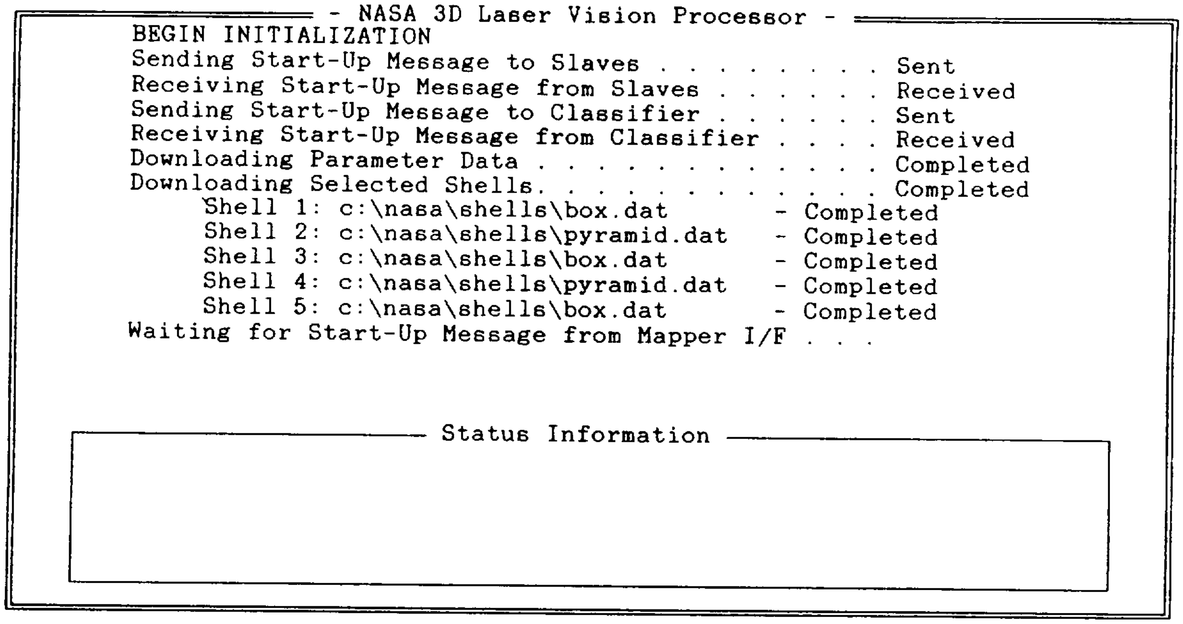
This executes a batch file "gn.bat" (gn stands for GO NASA) which boots up the host transputer and initializes the system to the default configuration. The program opens and reads the file "c:\SHELL\mission.lst", which contains the names of system parameter and shell description files. Three command line options are permitted by "gn.bat"; these are:

1) *C:\NASA>***gn /P filename.ext** : allows the user to designate a file other than "mission.lst" containing system parameter and shell description filenames.

2) *C:\NASA>***gn /D** : Turns on debug messages.

3) *C:\NASA>***gn /E** : Turns on edge display.

After starting "gn", the following screen will appear:



```
                    - NASA 3D Laser Vision Processor -
     BEGIN INITIALIZATION
     Sending Start-Up Message to Slaves . . . . . . . . . Sent
     Receiving Start-Up Message from Slaves . . . . . . . Received
     Sending Start-Up Message to Classifier . . . . . . . Sent
     Receiving Start-Up Message from Classifier . . . . . Received
     Downloading Parameter Data . . . . . . . . . . . . . Completed
     Downloading Selected Shells. . . . . . . . . . . . . Completed
          Shell 1: c:\nasa\shells\box.dat      - Completed
          Shell 2: c:\nasa\shells\pyramid.dat  - Completed
          Shell 3: c:\nasa\shells\box.dat      - Completed
          Shell 4: c:\nasa\shells\pyramid.dat  - Completed
          Shell 5: c:\nasa\shells\box.dat      - Completed
     Waiting for Start-Up Message from Mapper I/F . . .

                         Status Information
```

On the right side of the screen are status messages for each activity. These status messages describe the progress of the current process, and should an error occur, present an error code to the user. The error codes are defined in the file "display.inc", and are shown below.

Error Codes

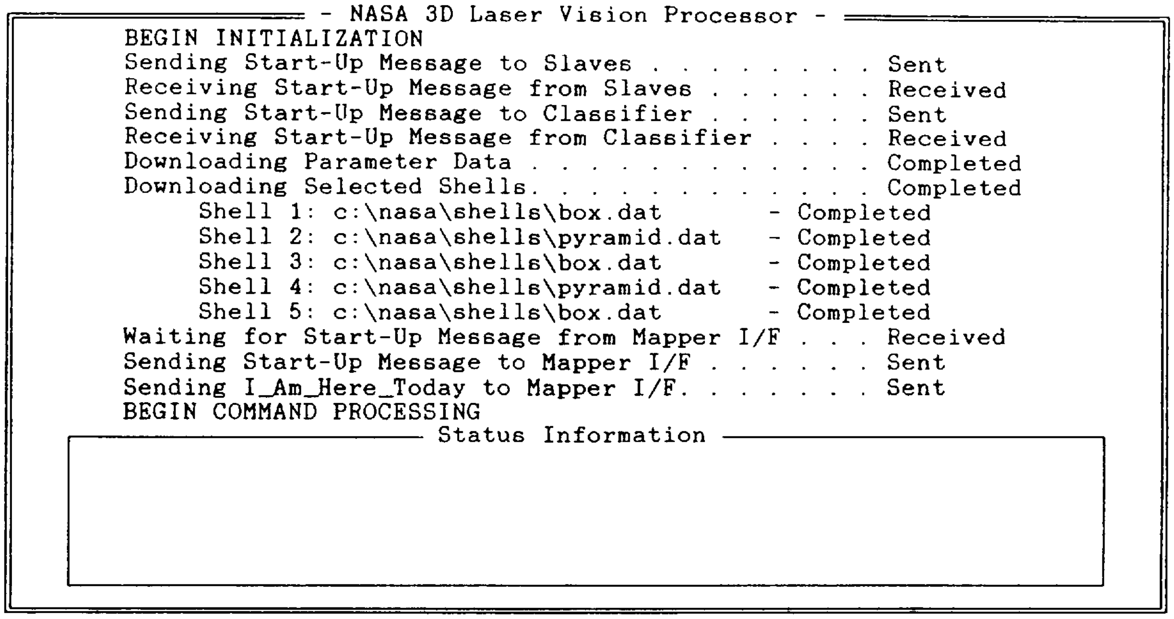
```
-- Mission File Related Error Codes
            Mission.File.Open.Fail                IS    64 (BYTE):
-- Parameter File Related Error Codes
            Parm.Open.Fail                        IS    65 (BYTE):
            Parm.Read.Fail                        IS    66 (BYTE):
            Bad.Parm.Record                       IS    65 (BYTE):
            Parm.Close.Fail                       IS    68 (BYTE):
-- Shell File Related Error Codes
            Shell.Open.Fail                       IS    69 (BYTE):
            Shell.Read.Fail                       IS    70 (BYTE):
            Shell.Bad.No.Polygons                 IS    71 (BYTE):
            Bad.Shell.Record                      IS    72 (BYTE):
            Shell.Close.Fail                      IS    73 (BYTE):
            Shell.Bad.Vertice.No                  IS    74 (BYTE):
            Shell.Vertice.Read.Fail               IS    75 (BYTE):
            Shell.Bad.No.Vertices                 IS    76 (BYTE):
            Shell.Bad.Polygon.No                  IS    77 (BYTE):
            Shell.Polygon.No.Read.Fail            IS    78 (BYTE):
            Shell.No.Polygons.Read.Fail           IS    79 (BYTE):
            Shell.Bad.Vertice.Coordinate          IS    80 (BYTE):
            Shell.No.Vertices.Read.Fail           IS    81 (BYTE):
            Shell.Slice.Height.Read.Fail          IS    82 (BYTE):
            Shell.Bad.Slice.Height                IS    83 (BYTE):
```

A box labeled "STATUS INFORMATION" is shown in the lower third of the startup display. This box is for program growth, and is not used at this time.

After completion of the system initialization, the following message appears:

"Waiting for Start-Up Message from Mapper I/F . . . "

At this time the user should initiate the Mapper control program. When Mapper has taken control of the system, the following additional messages will appear in the screen:



```
                - NASA 3D Laser Vision Processor -
       BEGIN INITIALIZATION
       Sending Start-Up Message to Slaves . . . . . . . . Sent
       Receiving Start-Up Message from Slaves . . . . . . Received
       Sending Start-Up Message to Classifier . . . . . . Sent
       Receiving Start-Up Message from Classifier . . . . Received
       Downloading Parameter Data . . . . . . . . . . . . Completed
       Downloading Selected Shells. . . . . . . . . . . . Completed
            Shell 1: c:\nasa\shells\box.dat       - Completed
            Shell 2: c:\nasa\shells\pyramid.dat   - Completed
            Shell 3: c:\nasa\shells\box.dat       - Completed
            Shell 4: c:\nasa\shells\pyramid.dat   - Completed
            Shell 5: c:\nasa\shells\box.dat       - Completed
       Waiting for Start-Up Message from Mapper I/F . . . Received
       Sending Start-Up Message to Mapper I/F . . . . . . Sent
       Sending I_Am_Here_Today to Mapper I/F. . . . . . . Sent
       BEGIN COMMAND PROCESSING
                         Status Information
```

The program then switches to the graphics screen.

Graphics Screen

After initialization, the display of the PC switches into the graphics mode, and the following screen appears:

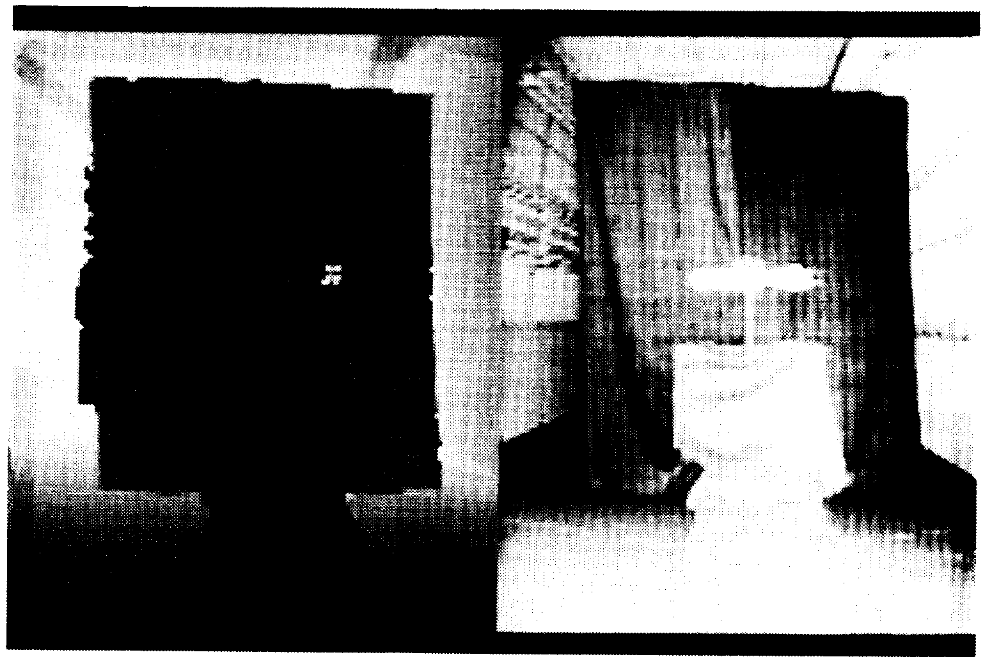


Figure 4. PC Display Range/Intensity

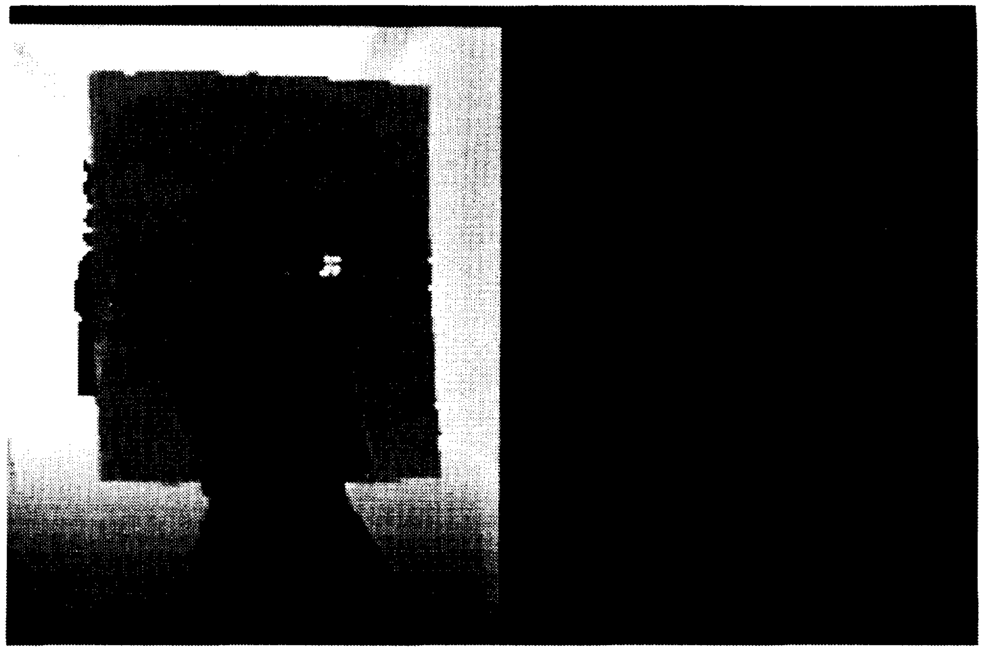


Figure 5. PC Display Range/Detected Objects

The screen is divided into four areas: range display, intensity display, command/debug message display, and hot key status display. These are described below.

Range Display:	Image range data is presented in the color map selected using the hot keys (see below).
Intensity Display:	Image intensity data is presented using the color map selected using the hot keys (see below).
Debug Display:	If the debug facility is active (either through the command line parameter /D or a hot key as explained below), debug messages will be scrolled in this area.

Hot Key Status: The upper right portion of the screen contains hot key status indicators. To toggle a hot key status, type the letter (not case sensitive) of the desired function shown below. The changed status will be valid for the next screen to be processed, not the current screen. The hot keys are listed below:

B - Toggle blob array display

D - Toggle debug display

E - Toggle edge array display

I - Toggle intensity array display

L? - Change display color map where:

L0	=	black & white, normal
L1	=	black & white, inverted
L2	=	sawtooth
L3	=	sawtooth
L4	=	sawtooth
L5	=	sawtooth
L6	=	gradual 3 colors
L7	=	random colors
L8	=	color spline
L9	=	sawtooth

R - Toggle range array display

3.2 Mission/Parameter Files

Execution of the NASA 3-D Laser Vision Processor requires that two files be present: The Mission File (default = MISSION.LST) and the parameter data file (PARMS.DAT or similar file designated by the Mission File).

3.2.1 Mission File

MISSION.LST contains the names of files which contain system parameters and object geometry descriptions. The sample file supplied with the software is shown below.

```
;
; Mission Test File
;
c:\nasa\shells\parms.dat ; Mission Parameters
;
; Mission Shell Files
;
c:\nasa\shells\box.dat ; Mission Shell
c:\nasa\shells\pyramid.dat
c:\nasa\shells\box.dat ; Mission Shell
c:\nasa\shells\pyramid.dat
c:\nasa\shells\box.dat ; Mission Shell
; end of file
```

A file of similar structure can be substituted for MISSION.LST through the use of the /P command line option described above. For instance, to instruct the program to use a file named RUN1.LST instead of MISSION.LST, type:

```
C:\NASA>gn /P run1.lst
```

RUN1.LST must contain, in this order, the name of a file containing program parameters similar in structure to PARMS.DAT (see below), and up to five target shell geometry files.

3.2.2 Parameter Data File

PARMS.DAT contains system parameters data such as Field of View (FOV), detection parameters such as object size and intensity thresholds, etc. The sample file supplied with the software is shown below.

```
;
; NASA Parameter Data
;
; System Parameters
;
60.0  ; hfov (degrees)
60.0  ; vfov (degrees)
0.0   ; beta_bias (degrees)
; Detect Parameters
200 ; MIN_INTEN_THR (0 to 255)
150 ; MAX_RANGE_THR (0 to 255)
6   ; MIN_OBJ_SIZE   was 2
4   ; STKADD_THR_RC
4   ; STKADD_THR_RR
;
; Track Parameters
;
20  ; MAXERR;
5   ; MTHRES
;
; Classify Parameters
;   TBD
; end of file
```

PARMS.DAT is the file name contained in the sample MISSION.LST file. If it is desired to use another file of similar structure rather than PARMS.DAT, the Mission File (i.e. MISSION.LST) must contain this filename as the first line which is not a comment.

3.3 Constants Files

Values for various parameters for functions such as screen background color, Mapper ID numbers, Mapper interface commands, etc., are read from files at compilation and become part of the executable file. Since these parameters cannot be changed at runtime, recompilation must be performed if changes are desired. Recompilation can be accomplished quite easily by using the batch file MAKEALL.BAT, located in the C:\NASA directory. After the desired alterations to the parameter files described below, typing "MAKEALL" recompiles all FORTRAN and OCCAM files and sets up the network. This facility allows fairly painless customization of the program parameters. The files containing these parameters are CONSTANT.INC and DISPLAY.INC.

The CONSTANT.INC file supplied with the software is shown in Appendix A. While any of these values can be modified, some values are "hard wired" into the source code and would require source code alterations to modify. These parameters are usually not of interest to the user. Parameters in this file which can easily be modified are those dealing with interface to the Mapper such as IDs, commands, messages, etc. These parameters are found at the beginning of the file and extend up to (but not including) the section labeled "Master to Slave Protocol".

The second constant file is DISPLAY.INC, which concerns screen colors and pixel coordinates, error codes, etc. Any of these parameter values can be modified with no alterations to the source code required. The DISPLAY.INC file supplied with the software is given in Appendix B.

Shell Files

Appendix C presents several examples of shell files compatible with the NASA 3-D Laser Vision Processor. The coordinate system used in these files is shown in Figure 6. The attitude angle definitions are shown in Figure 7.

3D MAPPER

COORDINATE SYSTEM DEFINITION

- All positions relative to Mapper boresight with respect to a person standing behind Mapper

+X	-	Axis in front of Mapper
+Y	-	Axis to the right
+Z	-	Axis below Mapper

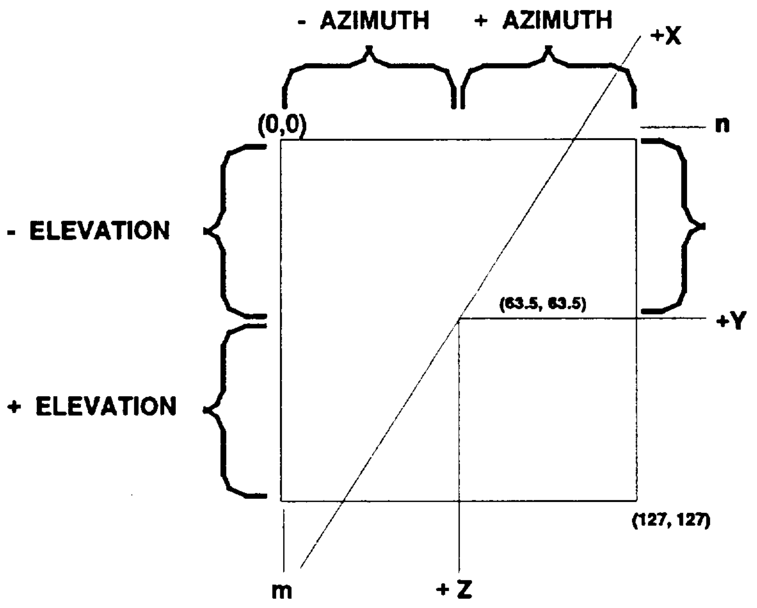
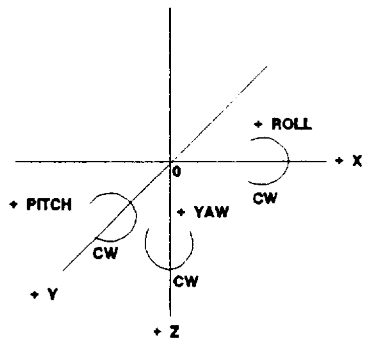


Figure 6. Odetics 3D Mapper Image Description

TARGET ATTITUDE



ROTATION SEQUENCE: YAW, PITCH, ROLL

All positive senses in CW direction with respect to observer at 0

Figure 7. Target Attitude Definition

APPENDIX A - CONSTANT.INC LISTING

CONSTANT.INC

```
--
-- LORAL and NASA ID Codes
--
VAL Laser.Processor         IS #0510 (INT32): -- LORAL Id
VAL Vision.Control          IS #0500 (INT32): -- NASA Id
VAL Range.Control           IS #0504 (INT32):
VAL Goode.Ranger            IS #0507 (INT32):
VAL Display                 IS #0502 (INT32): -- RGB Monitor on
                                                  VC
VAL Pcif                    IS #0501 (INT32): -- File and Pics
VAL Vision.Console          IS #0505 (INT32): -- Debug Messages
--
-- NASA Source ID Codes
--
VAL Mapper                  IS 10 (INT32):
--
-- NASA Command Constants
--
-- General/Phase Control Input Commands
--
VAL Null.Function           IS #0000 (INT32):
VAL Enable                  IS #1000 (INT32):
VAL Disable                 IS #1001 (INT32):
VAL Reset                   IS #1002 (INT32):
VAL Safe                    IS #1003 (INT32):
VAL Idle                    IS #1004 (INT32):
VAL Resume                  IS #1005 (INT32):
```

```
VAL Set.Time                        IS #100B (INT32):
VAL Target.Acq.Manual               IS #100C (INT32):
VAL Target.Acq.Auto                 IS #100D (INT32):
VAL Blob.Detect                     IS #100F (INT32):
VAL Blob.Track                      IS #1010 (INT32):
VAL Determine.Grasp                 IS #1011 (INT32):
VAL Monitor.Grasp                   IS #1012 (INT32):
VAL Are.You.Here.Today              IS #1015 (INT32):
VAL Send.Health.Status              IS #1016 (INT32):
VAL Stop.Blob.Track                 IS #1019 (INT32):
VAL Use.Mapper.Scan.Direction       IS #101F (INT32):
--
-- Data/Control Input/Output Commands
--
VAL Sending.Blobs                   IS #1100 (INT32):
VAL No.Blobs.Found                  IS #1108 (INT32):
VAL No.Grasp.Regions                IS #1109 (INT32):
VAL Health.Status                   IS #110A (INT32):
VAL I.Give.Up                       IS #110B (INT32):
VAL I.Am.Here.Today                 IS #110C (INT32):
VAL Sending.Raw.Mapper.Data         IS #110F (INT32):
VAL Debug.Message                   IS #1111 (INT32):
--
-- Odetics Hardware Commands
--
VAL Mapper.Xfer.Frame.Of.Data       IS #1306 (INT32):
--
-- PCIF Interface Commands
--
VAL Pcif.Mapper.Results             IS #1504 (INT32):
```

```
VAL Pcif.Start.Sending.Visual.Results      IS #1509 (INT32):
VAL Pcif.Stop.Sending.Visual.Results       IS #150A (INT32):
--
-- NASA Start Up Messages
--
VAL []BYTE Good.Day.FTGWN                  IS "GOOD DAY FROM THE GREAT WHITE
                                           NORTH?!":
VAL []BYTE Good.Day.Eh                     IS "GOOD DAY, EH?!":
--
-- Unknown Data Value Constants
--
VAL Data.Null.Int                          IS #FFFFFFFF (INT32):
VAL REAL32 Data.Null.Float                 RETYPES #FFFFFFFF (INT32):
--
-- Health Status Values
--
VAL Nominal                                IS  1 (INT32):
VAL Abnormal                               IS -1 (INT32):
VAL Not.Here                               IS  2 (INT32):
--
-- Scan Direction Definition
--
VAL Up.Scan                                IS  0  (INT32):  VAL  Down.Scan
                                           IS  1 (INT32):
VAL Any.Scan                               IS  2 (INT32):
----------------------------------------------------------------


-- Master to Slave protocol
--
----------------------------------------------------------------
```

```
PROTOCOL MS
          CASE
          Startup.Tag;                  BYTE
          Parameter.Tag;                INT32::[]INT32
          Shell.Tag;                    INT32::[]INT32
          Operate.Tag;                  INT32;INT32;INT32::[]BYTE
          Odetics.Tag;                  INT32;INT32::[]BYTE
          Object.Tag;                   INT32::[]INT32
          Detect.Tag;                   INT32::[]INT32;
                                        INT32::[]INT32;INT32::[]INT32;
                                        INT32::[]INT32;INT32::[]INT32
          Extent.Tag;                   INT32;INT32;
                                        INT32;INT32;
                                        INT32;INT32
          Disable.Tag
          Set.Time.Tag;                 INT32
          Edge.Toggle.Tag;              BOOL
:
--------------------------------------------------------------------
-- Classifier to Master Protocol
--
--------------------------------------------------------------------
PROTOCOL CM
          CASE
          Init.Tag;                     BYTE
          Blob.Tag;                     INT32;INT32;INT32;INT32;INT32;INT32;
                                        INT32;INT32;
                                        REAL32;REAL32;REAL32;REAL32;REAL32;
                                        REAL32;
                                        REAL32;REAL32;REAL32;REAL32;REAL32;
```

```
                        REAL32;
                        REAL32;REAL32;REAL32;REAL32;REAL32;REAL32
    Edge.Tag;           INT32::[]INT32
:
--------------------------------------------------------------
--
-- Detect Command Tags
--
--------------------------------------------------------------
VAL Detect.Reset.Tag        IS 1 (INT32):
VAL Detect.Parms.Tag        IS 2 (INT32):
VAL Detect.Range.Tag        IS 3 (INT32):
VAL Detect.Intensity.Tag    IS 4 (INT32):
VAL Detect.Go.Tag           IS 5 (INT32):
--------------------------------------------------------------
```

```
--
-- Track Command Tags
--
------------------------------------------------------------
VAL Track.Reset.Tag          IS 1 (INT32):
VAL Track.Init.Tag           IS 2 (INT32):
VAL Track.Pretrack.Tag       IS 3 (INT32):
VAL Track.Object.Tag         IS 4 (INT32):
VAL Track.Last.Tag           IS 5 (INT32):
VAL Track.Output.Tag         IS 6 (INT32):
------------------------------------------------------------
--
-- Classify Command Tags
--
------------------------------------------------------------
VAL Classify.Reset.Tag       IS 1 (INT32):
VAL Classify.Parms.Tag       IS 2 (INT32):
VAL Classify.Shell.Tag       IS 3 (INT32):
VAL Classify.Range.Tag       IS 4 (INT32):
VAL Classify.Object.Tag      IS 5 (INT32):
VAL Classify.Blob.Tag        IS 6 (INT32):
VAL Classify.Edge.Tag        IS 7 (INT32):
------------------------------------------------------------
--
-- Parameter Tags
--
--      Parameter Tags are Indexed by the Following Values.
--              To ADD Parameters, Change the Index Below and Add the
--              Parameters to the Parameter Files.
--
```

```
--      Change the Variable Max.Parms below to be Maximum Number of
--              Parameters + 1. The +1 is because the Parameter Count is
                Stored in Parameter[0].
-----------------------------------------------------------------
--
VAL Max.Parms                   IS 11 (INT):
--
-- System Parameter Tags
--
VAL Parameter.Count             IS  0 (INT):
VAL Horz.FOV                    IS  1 (INT):
VAL Vert.FOV                    IS  2 (INT):
VAL Beta.Bias                   IS  3 (INT):
--
-- Detect Parameter Tags
--
VAL Min.Inten.Thr               IS  4 (INT):
VAL Max.Range.Thr               IS  5 (INT):
VAL Min.Obj.Size                IS  6 (INT):
VAL Stkadd.Thr.Rc               IS  7 (INT):
VAL Stkadd.Thr.Rr               IS  8 (INT):
--
-- Track Parameter Tags
--
VAL Max.Err                     IS  9 (INT):
VAL Mth.Res                     IS 10 (INT):
--
-- Classify Parameter Tags
--
--   T.B.D.
```

```
--
-- Local Constants
--
VAL Blob.Size                   IS    26 (INT):
VAL Buffer.Size                 IS 65536 (INT):
VAL Header.Size                 IS   208 (INT):
VAL Image.Size                  IS (128 * 128) * 2 (INT):
VAL Intensity.Size              IS Image.Size / 2 (INT):
VAL Max.No.Obj                  IS  1000 (INT):
VAL Max.Shells                  IS     5 (INT):
VAL Max.Shell.Parms             IS  1000 (INT):
VAL No.of.Cols                  IS   128 (INT32):
VAL No.of.Rows                  IS   128 (INT32):
VAL Object.Size                 IS Image.Size / 2 (INT):
VAL Parameter.Scale             IS 32768.0 (REAL32):
VAL Range.Size                  IS Image.Size / 2 (INT):
VAL Shell.Coordinate.Scale      IS 32768.0 (REAL32):
VAL Shell.Slice.Scale           IS 32768.0 (REAL32):
--
VAL Executive                   IS TRUE:
VAL Subordinate                 IS FALSE:
--
```

APPENDIX B - DISPLAY.INC LISTING

DISPLAY.INC

```
#INCLUDE "hostio.inc"
--
-- Local Constants
--
VAL Background.Color           IS    0 (INT32):
VAL Bell                       IS '*#07':
VAL Comment                    IS ';' (BYTE):
VAL []BYTE Erase.Display       IS "*#1B[2J":
VAL Keyboard.Delay             IS 1000 (INT):
VAL Line.Color                 IS  255 (INT32):
VAL Lrcol                      IS   39 (INT32):
VAL Lrrow                      IS   24 (INT32):
VAL []BYTE Mission.List        IS "\nasa\shells\mission.lst":
VAL []BYTE Set.Color           IS "*#1B[44;33;1m":
VAL []BYTE Set.Mode            IS "*#1B[=3h":
VAL Ulcol                      IS    0 (INT32):
VAL Ulrow                      IS   17 (INT32):
VAL Void.Color                 IS    0 (INT32):
--
-- Initialization Border Coordinates
--
VAL Border.x1                  IS    2 (INT):
VAL Border.y1                  IS    1 (INT):
VAL Border.x2                  IS   79 (INT):
VAL Border.y2                  IS   25 (INT):
--
```

```
-- Status Box Display Coordinates
--
VAL Status.x1                   IS   6 (INT):
VAL Status.y1                   IS  18 (INT):
VAL Status.x2                   IS  75 (INT):
VAL Status.y2                   IS  24 (INT):
--
-- Error Codes
--
-- Mission File Related Error Codes
VAL Mission.File.Open.Fail      IS  64 (BYTE):
-- Parameter File Related Error Codes
VAL Parm.Open.Fail                   IS  65 (BYTE):
VAL Parm.Read.Fail                   IS  66 (BYTE):
VAL Bad.Parm.Record                  IS  65 (BYTE):
VAL Parm.Close.Fail                  IS  68 (BYTE):
-- Shell File Related Error Codes
VAL Shell.Open.Fail             IS  69 (BYTE):
VAL Shell.Read.Fail             IS  70 (BYTE):
VAL Shell.Bad.No.Polygons       IS  71 (BYTE):
VAL Bad.Shell.Record            IS  72 (BYTE):
VAL Shell.Close.Fail            IS  73 (BYTE):
VAL Shell.Bad.Vertice.No        IS  74 (BYTE):
VAL Shell.Vertice.Read.Fail     IS  75 (BYTE):
VAL Shell.Bad.No.Vertices       IS  76 (BYTE):
VAL Shell.Bad.Polygon.No        IS  77 (BYTE):
VAL Shell.Polygon.No.Read.Fail  IS  78 (BYTE):
VAL Shell.No.Polygons.Read.Fail      IS  79 (BYTE):
VAL Shell.Bad.Vertice.Coordinate     IS  80 (BYTE):
VAL Shell.No.Vertices.Read.Fail IS  81 (BYTE):
```

```
VAL Shell.Slice.Height.Read.Fail IS   82 (BYTE):
VAL Shell.Bad.Slice.Height       IS   83 (BYTE):
--
VAL Server.EOF                   IS  128 (BYTE):
--
VAL Option.Strings IS ["D", "E", "P"]:
VAL Option.Parameters.Required              IS [spopt.never,spopt.never,spopt.always]:
VAL Max.Options                             IS 3 (INT):
VAL Is.Debug                                IS 0 (INT):
VAL Is.Edge                                 IS 1 (INT):
VAL Is.Parameter                            IS 2 (INT):
```

APPENDIX C - SHELL GENERATION

The following section describes the creation of target shell surface models for use by the 3-D Laser Radar Vision Processor System. The processor system uses these models for comparison against live objects received from the laser radar sensor. Each model consists of a 3-D surface shell along with the dimensions of the primary plane. These surface shells, which are polygonal representations of the outer surface of each target, are generated from scale drawings or photographs. They are stored in a Computer Aided Design (CAD) representation in the processor memory. In this format, each surface shell is represented by a number of vertices in a local coordinated system and the four-sided polygons defined by these vertices. The advantage of this is that a complex target model (i.e., on the order of a couple hundred polygons) can be stored in a relatively small amount of memory.

Each model is generated from the dimensional data of actual targets. The program does not scale the model dimensional to fit the target. That is, a generic wrench or box cannot be stored as a surface shell. For each dimensional size of the target, a separate model is generated to describe it. Even though having several target models at various sizes might seem like a processor burden, the processing time is not overall affected since the algorithm prescreens each of the target models rejecting any of them which are inconsistent in size dimension.

The classification portion of the algorithm assumes that when a target model is generated it conforms to several requirements. When the target model is in an unrotated state (yaw, pitch and roll are zero degrees), the primary plane is situated such that the sensor coordinated system's (SCS) x value is equal to a constant and that the longest axis of this plane is parallel the SCS y axis. The orientation procedure assumes that an object has a primary plane and that the orientation of the object is such that the orientation of the primary plane is the orientation of the object. When the object is rotated, the rotation occurs about the object center of mass. A local coordinate system, called the object coordinate system (OCS) is defined as a translated SCS with origin at the object center of

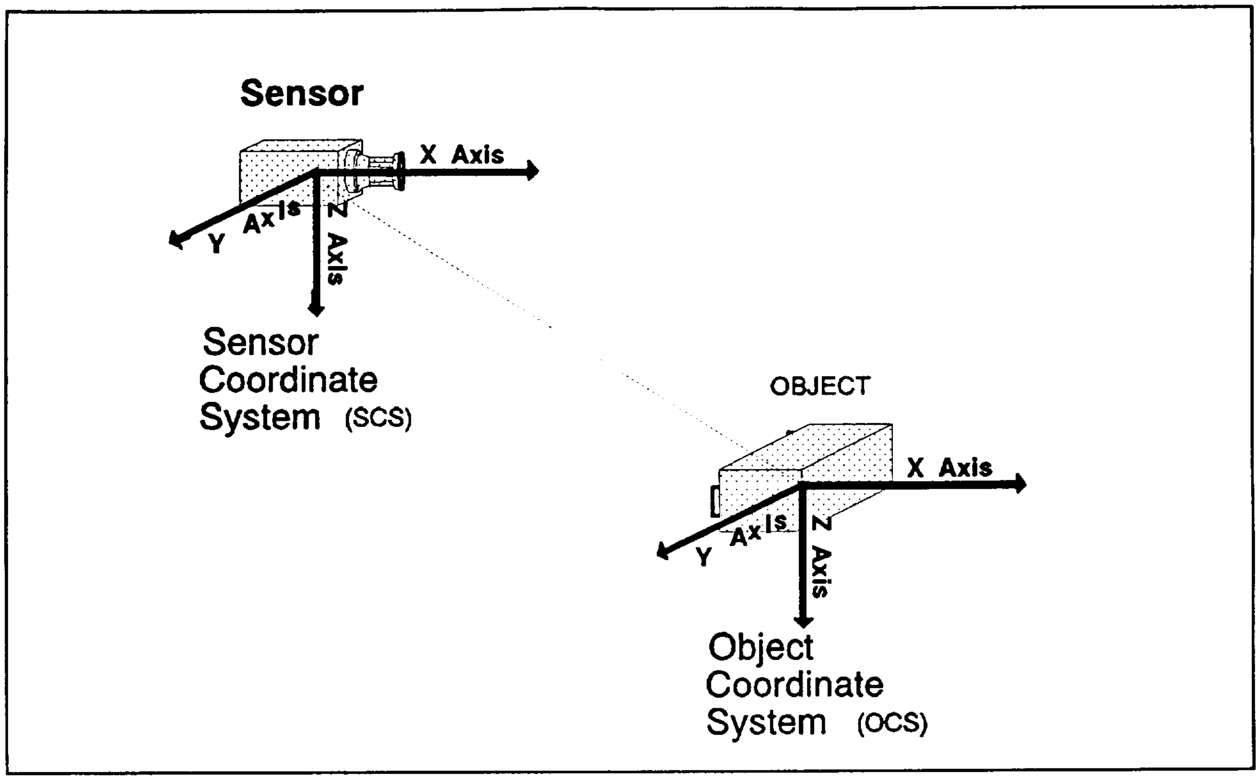

Sensor
X Axis
Y Axis
Z Axis
Sensor Coordinate System (SCS)
OBJECT
X Axis
Y Axis
Z Axis
Object Coordinate System (OCS)

Figure C-1. Coordinate System Definition

mass. Figure C-1 illustrates this coordinate system. When the object is rotated, the order of rotation will bedefined to be in the order of yaw, then pitch and then roll. Yaw, pitch and roll angles are defined to be the clockwise rotation about the z axis, y axis, and x axis, respectively. This is illustrated by Figure C-2. To limit processing time, a current requirement of the algorithm is that the primary plane must always be in the field of view.

The file format of a target surface shell model is given by Table C-1. An example of using this format on an 'ORU1' (box) model (see Figure C-3) with the primary plane defined to be side 5 is illustrated by Table C-2.

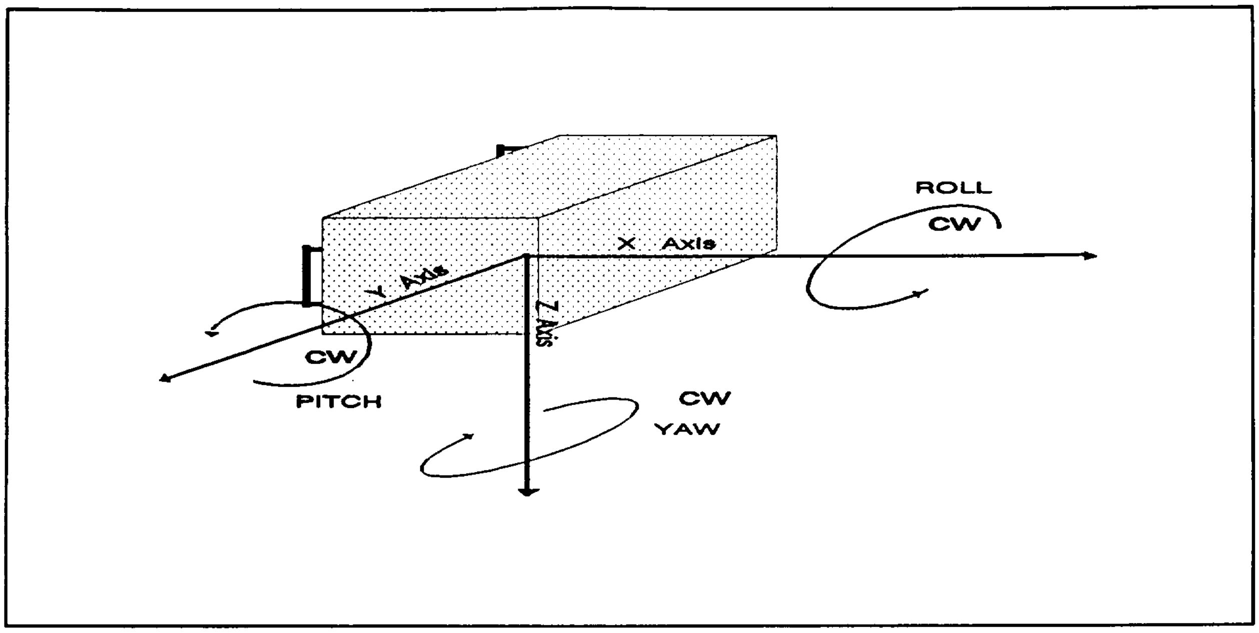


Figure C-2. Target Attitude Definition

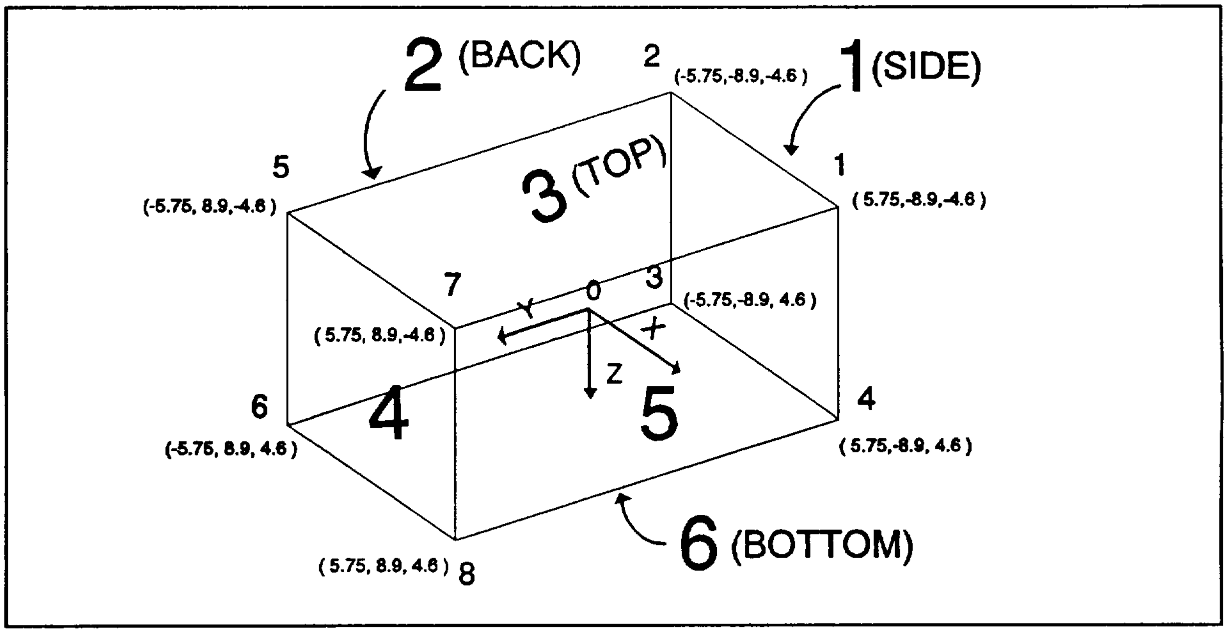


Figure C-3. ORU1 (BOX) Surface Shell Model Definition

Table C-1. SHELL MODEL FILE FORMAT

```
NP
PN              NV
VN1             VN2          VN3          NV4
PN              NV
VN1             VN2          VN3          VN4
.               .            .            .
.   Repeats for all polygons .
.               .            .            .
.               .            .            .
.               .            .            .
.               .            .            .
PN              NV
VN1             VN2          VN3          VN4
NVERT
CVN1            CVN1  X       Y             Z
CVN2            CVN2  X       Y             Z
.               .            .            .
    Repeats for all vertices .
.               .            .            .
.               .            .            .
.               .            .            .
.               .            .            .
CVNn            CVNn  X       Y             Z
SL              SW           SH
```

WHERE:	NP =	The number of 4-sided polygons in the model
	PN =	The number of an individual polygon
	NV =	The number of vertices in the following polygons (fixed at four for our models)
	VNn =	The number of an individual vertex in the polygon numbered in the preceding line
	NVERT =	The number of vertices in the model
	CVNn =	The number of an individual vertex
	X =	The X coordinate of the vertex number on this line
	Y =	The Y coordinate of the vertex number on this line
	Z =	The Z coordinate of the vertex number on this line
	SL =	Primary plane length
	SW =	Primary plane width
	SH =	Depth of target normal to primary plane

Vertex number (VNn) are entered on a line from left to right as read from the model in a clockwise (CW) direction with an outward normal to the polygon plane formed by the 4 vertices.
All data is in ASCII coding.
Spacing of data in columns is very important. All data are in a 15-character field.

Table C-2. Example file format - ORU1 (Box)

```
8
6
1        4
1        2        3        4
2        4
2        5        6        3
3        4
1        7        5        2
4        4
5        7        8        6
5        4
7        1        4        8
6        4
3        6        8        4
8
1        1  5.7500000   -8.9000000   -4.60000000
2        2 -5.7500000   -8.9000000   -4.60000000
3        3 -5.7500000   -8.9000000    4.60000000
4        4  5.7500000   -8.9000000    4.60000000
5        5 -5.7500000    8.9000000   -4.60000000
6        6 -5.7500000    8.9800000    4.60000000
7        7  5.7500000    8.9000000   -4.60000000
8        8  5.7500000    8.9000000    4.60000000
17.8000      9.2000       11.5000
0.0000       0.0000       0.0000
```